AF
ww

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Appellants:	Altman et al.	Examiner: W. Wood
Serial No:	09/637,078	Group Art Unit: 2193
Filed:	August 11, 2000	Docket: YOR92000-0415US1 (8728-407)

For: **METHOD AND APPARATUS FOR PROFILING COMPUTER PROGRAM EXECUTION**

APPEAL BRIEF

This is an Appeal from the Office Action mailed October 6, 2006 rejecting claims 1, 3-16, 18-30 and 32-42 in which prosecution was reopened in view of the Appeal Brief filed on July 14, 2005. Applicants reinstated the Appeal Brief by Notice of Appeal filed on January 9, 2006 and submit this Appeal Brief in furtherance of the Appeal.

Appeal from Group 2193

F. Chau & Associates, LLC
130 Woodbury Road
Woodbury, NY 11797
TEL: (516) 692-8888
FAX: (516) 692-8889
Attorneys for Appellants

TABLE OF CONTENTS

1. Real Party in Interest

The real party in interest is INTERNATIONAL BUSINESS MACHINES CORPORATION, the assignee of the entire right, title and interest in and to the subject application by virtue of an assignment of record.

2. Related Appeals and Interferences

None.

3. Status of Claims

Claims 1, 3-16, 18-30 and 32-42 are pending, stand rejected and are under appeal.

A copy of the claims 1, 3-16, 18-30 and 32-42 as pending is presented in the Appendix.

4. Status of Amendments

Claims 1, 3-16, 18-30 and 32-42 were not amended after Final Rejection.

5. Summary of Claimed Subject Matter

For purposes of illustration, the invention of claims 1, 23, and 40, will be described with reference to the exemplary FIGs. and corresponding text of Appellants' Specification (hereinafter, Spec.), for example, but nothing herein shall be deemed as a limitation on the scope of the invention. In general, the claimed inventions are directed to a method and apparatus for profiling computer program execution.

Claim 1 recites:

A method for profiling computer program executions in a computer processing system having a processor and a memory hierarchy, comprising the steps of:

executing a computer program;

storing, in a memory array, profile counts for a plurality of events associated with the execution of the computer program, the memory array being separate and distinct from the memory hierarchy so as to not perturb normal operations of the memory hierarchy;

selecting a at least one of the plurality of events for profiling;

updating the profile counts for only the selected events

assisting compilation and optimization of the computer program, based upon the selected profile counts stored in the memory array.

A method for profiling computer program executions in a computer processing system having a processor and a memory hierarchy is claimed in claim 1. The method includes the following steps. A computer program is executed. (Spec., p. 19, line 7; p. 20, line 17). Profile counts are stored in a memory array. The profile counts are for a plurality of events associated with the execution of the computer program. The memory array is separate and distinct from the memory hierarchy so as to not perturb normal operations of the memory hierarchy. (Spec., p. 14, lines 8-13; Figure 1, profile matrix 100; Figure 5, profile matrix 100). At least one of the plurality of events is selected for profiling. (Spec., p. 19, lines 11-14; Figure 3, step 304; Figure 4, step 404). The profile counts are updated for only the selected events. (Spec., p. 19, lines 20-21; p. 17, lines 15-22). Compilation and optimization of the computer program is assisted, based upon the selected profile counts stored in the memory array. (Spec., p. 22, line 22-p. 26, line 14; p. 28, line 19-p.29, line 8).

Claim 23 recites:

An apparatus for profiling computer program executions in a computer processing system having a processor and a memory hierarchy, the apparatus comprising:

a memory array adapted to store profile counts for events associated with execution of the computer program, said memory array being separate and distinct from the memory hierarchy so as to not perturb normal operations of the memory hierarchy;

a controller adapted to select the events for profiling and to update the profile counts of the selected events stored in said memory array; and

a scaling circuit adapted to scale the profile counts to prevent profile information overflow;

wherein the computer processing system assists compilation of the computer program, based upon the profile counts stored in the memory array.

An apparatus for profiling computer program executions in a computer processing system having a processor and a memory hierarchy is claimed in claim 23. The apparatus includes the following. A memory array is provided that is adapted to store profile counts for events associated with execution of the computer program. The memory array is separate and distinct from the memory hierarchy so as to not perturb normal operations of the memory hierarchy. (Spec., p. 14, lines 8-13; Figure 1, profile matrix 100; Figure 5, profile matrix 100). A controller is provided that is adapted to select the events for profiling and to update the profile counts of the selected events stored in said memory array. (Spec., p. 19, lines 11-23; Figure 3, steps, 304, 310; Figure 4, step 404, 410). A scaling circuit is provided that is adapted to scale the profile counts to prevent profile information overflow. (Spec., p. 15, lines 14-24). The computer processing system assists compilation of the computer program, based upon the profile counts stored in the memory array. (Spec., p. 22, line 22-p. 26, line 14; p. 28, line 19-p.29, line 8).

Claim 40 recites

A method for profiling computer program executions in a computer processing system having a processor and a memory hierarchy, comprising the steps of:

executing a computer program;

storing, in a memory array, a plurality of event-specific profile counts, each associated with an event associated with the execution of a path of the computer program, the memory array being separate and distinct from the memory hierarchy so as to not perturb normal operations of the memory hierarchy;

selecting at least one of the plurality of event-specific profile counts for profiling the path of the computer program; and

A method for profiling computer program executions in a computer processing system having a processor and a memory hierarchy is claimed in claim 40. The method includes the following steps. A computer program is executed. (Spec., p. 19, line 7; p. 20, line 17). Event-specific profile counts are stored in a memory array. Each profile count is associated with an event associated with the execution of a path of the computer program. The memory array is separate and distinct from the memory hierarchy so as to not perturb normal operations of the memory hierarchy. (Spec., p. 14, lines 8-13; Figure 1, profile matrix 100; Figure 5, profile matrix 100). At least one of the plurality of event-specific profile counts is selected is selected for profiling the path of the computer program. (Spec., p. 19, lines 11-14; Figure 3, step 304; Figure 4, step 404). If at least one of the selected event-specific profile counts has exceeded a predefined threshold, the portions of the computer program associated with the event-specific profile counts are optimized more aggressively than other portions of the computer program. (Spec., p. 22, lines 2-8; p. 22, line 22-p. 26, line 14; p. 28, line 19-p.29, line 8).

6. Grounds of Rejection to be Reviewed on Appeal

A. Claims 1, 4-8, 11-13, 16, 22, and 38 stand rejected under 35 U.S.C. § 102(e) as being unpatentable over Krishnaswamy et al. (U.S. Patent No. 6.622,300) (hereinafter "Krishnaswamy").

B. Claims 1, 4-8, 11-13, 16, 22, and 38 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Krishnaswamy in view of Klapproth et al. (U.S. Patent No. 5,590,354) (hereinafter "Klapproth").

C. Claims 3, 9-10, 23-30, 32-34, 37, and 39 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Krishnaswamy in view of "Dictionary of Computing" (hereinafter "Dictionary").

D. Claims 3, 9-10, 23-30, 32-34, 37, and 39 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Krishnaswamy in view of Klapproth in view of Dictionary.

E. Claims 40 and 42 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Krishnaswamy in view of Chang et al. "Using Profile Information to Assist Classic Code Optimizations" (hereinafter "Chang")..

F. Claims 40 and 42 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Krishnaswamy in view of Klapproth in view of Chang.

G. Claims 14-15 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Krishnaswamy in view of Record et al. (U.S. Patent No. 5,355,484) (hereinafter "Record")

H. Claims 14-15 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Krishnaswamy in view of Klapproth in view of Record.

I. Claims 18-19 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Krishnaswamy in view of Altman et al. "DAISY: Dynamic Compilation for 100% Architectural Compatibility" (hereinafter "Altman")

J. Claims 18-19 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Krishnaswamy in view of Klapproth in view of Altman.

K. Claims 20-21 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Krishnaswamy in view of Altman in view of Chang.

L. Claims 20-21 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Krishnaswamy in view of Klapproth in view of Altman in view of Chang.

M. Claims 35-36 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Krishnaswamy in view of Dictionary in view of Record.

N. Claims 35-36 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Krishnaswamy in view of Klapproth in view of Dictionary in view of Record.

O. Claim 41 stands rejected under 35 U.S.C. § 103(a) as being unpatentable over Krishnaswamy in view of Chang in view of Altman.

P. Claim 41 rejected under 35 U.S.C. § 103(a) as being unpatentable over Krishnaswamy in view of Klapproth in view of Chang in view of Altman.

7. **Arguments**

A. **The Teachings of Krishnaswamy Do Not Support the Anticipation Rejections.**

For a claim to be anticipated under 35 U.S.C. § 102, all elements of the claim must be found in a single prior art reference (see, e.g., Scripps Clinic & Research Found. v. Genentech Inc., 927 F.2d 1565, 1576, 18 U.S.P.Q.2d. 1001, 1010 (Fed. Cir. 1991)). The identical invention must be shown in as complete detail as is contained in the claim. (See MPEP § 2131). The single prior art reference must disclose all of the elements of the claimed invention functioning essentially in the same manner (see, e.g., Shanklin Corp. v. Springfield Photo Mount Corp, 521 F.2d 609 (1st Cir. 1975)).

Here, Applicants respectfully assert that Krishnaswamy is legally deficient to establish prima facie case of anticipation against any of claims 1, 4-8, 11-13, 16, 22, and 38. At the very least, Krishnaswamy does not anticipate independent claim 1 for the following reasons.

(i). **Krishnaswamy fails to teach or suggest "selecting at least one of the plurality of events for profiling," as claimed in claim 1.**

Claim 1 claims, *inter alia*, "selecting at least one of the plurality of events for profiling." The Examiner admits on page 3 of the Final Office Action (Paper no. 112904) that "Krishnaswamy's background did not explicitly state selected events." The Examiner then cites col. 6, lines 21-30 of Krishnaswamy as demonstrating "selecting at least one of the purality of events for profiling. In the Advisory Action mailed on April 6, 2005, the Examiner specifically points to the word "programmable" in the phrase "programmable to count events like," as disclosed in col. lines 24-28 of Krishaswamy. In the Office action mailed on October 6, 2005, the examiner again cites col. lines 24-28 of Krishaswamy. The

Examiner seems to assume that the statement "*programmable* to count events like" necessarily teaches or suggests "*selecting* at least one of the plurality of events for profiling." The term "programmable" does not necessarily imply a particular function. In particular, the term "programmable" does not imply that a selection of events is necessarily accomplished. For example, the counters for Krishnaswamy can be programmed to count events A, B and C. However, that the counters can be programmed to count events A, B and C does not necessarily imply that only A can be selected, or that only A and B can be selected, or any of a variety of combinations for that matter. Basically, the Examiner is contending that the word "programmable" can be broadly interpreted to mean any function under the sun. Clearly, such a misunderstanding is without merit, facially unreasonable, and cannot be allowed.

Because Krishnaswamy neither teaches nor suggests each and every element of claim 1, it is respectfully asserted that no *prima facie* case of anticipation of claim 1 has been made out. Claims 4-8, 11-13, 16, 22, and 38 are patentable over Krishnaswamy at least by virtue of their dependance from claim 1. Accordingly, the rejection of claims 1, 4-8, 11-13, 16, 22, and 38 should be reversed.

(ii). Krishnaswamy fails to teach or suggest "updating the profile counts for only the selected events" and "assisting compilation and optimization of the computer program, based upon the selected profile counts stored in the memory array," as claimed in claim 1.

Because Krishnaswamy does not teach or suggest "selecting at least one of the plurality of events for profiling," as shown in in part (B)(i) above, it logically follows that Krishaswamy does not disclose "updating the profile counts *for only the selected events*"

and "assisting compilation and optimization of the computer program, *based upon the selected profile counts* stored in the memory array."

Because Krishnaswamy neither teaches nor suggests each and every element of claim 1, it is respectfully asserted that no *prima facie* case of anticipation of claim 1 has been made out. Claims 4-8, 11-13, 16, 22, and 38 are patentable over Krishnaswamy at least by virtue of their dependance from claim 1. Accordingly, the rejection of claims 1, 4-8, 11-13, 16, 22, and 38 should be reversed.

(iii). Krishnaswamy fails to teach or suggest "storing, in a memory array, profile counts for a plurality of events associated with the execution of the computer program, the memory array being separate and distinct from the memory hierarchy so as to not perturb normal operations of the memory hierarchy," as claimed in claim 1.

Claim 1 claims, *inter alia*, "storing, in a memory array, profile counts for a plurality of events associated with the execution of the computer program, the memory array being separate and distinct from the memory hierarchy so as to not perturb normal operations of the memory hierarchy".

In the Office Action mailed on October 6, 2006, the examiner states that "[t]he passages and figure illustrate a separate and distinct memory that is not perturbed by the profile" (column 5, lines 35-45, column 6, lines 34-36, and elements 70, 100, 160, and 170 of figure 2). However, claim 1 discloses storing profile data in a separate memory so as not to perturb the normal operations of the memory hierarchy.

Applicants can only assume that the examiner meant that the passages and figure illustrate a separate and distinct memory for the reason of not perturbing the normal operations of the memory hierachy. However, Krishnaswamy only discloses memories that are part of the memory hierachy such as a memory 70, permanent storage 160, and a

removable media unit 170 (column 4, lines 23-41). Since Krishnaswamy only discloses one memory, it is assumed that memory 70 is main memory. Memory hierachy, although well known in the art, is defined as registers, cache, main memory, secondary storage (disks), offline storage(tapes). See Edwin D. Reilly, Concise Encyclopedia of Computer Science, pp. 504, fig. 1 (2004). There is no mention in Krishnaswamy of a separate memory outside the memory hierarchy, just main memory (memory 70), disk storage (permanent storage 160), and offline disk storage (removable media unit 170).

The examiner further states that , "[i]f the profile data is stored in the kernel space then the shared user memory is considered the separate memory". The examiner seems to suggest that if the profile data is stored in the kernel space, then the shared user memory is a memory array separate and distinct from the memory hierarchy so as not to perturb the normal operations of the memory hierachy. However fig. 2 reveals that the shared user memory is located within main memory 70, and main memory is in the memory hierachy. Futhermore, shared user memory contains the very shared user processes that one would want to profile. So if profile data was stored in shared user memory, it would have to peturb the normal operations of the memory hierachy. In fact, it appears that Krishnaswamy discloses storing profile data in shared user memory and a streamlined version of profile data in kernel memory(column 5, lines 15-21). However, figure 2 reveals that the kernel memory is also located within main memory 70.

The examiner continues in the Office Action to suggest that "at the very least the separate and distinct memory is the removable memory unit 170 and the permanent storage 160." However, as discussed above, a removable media unit is the same as offline storage,

and permanent storage is the same as disk storage, all firmly planted within the memory hierarchy.

Because Krishnaswamy neither teaches nor suggests each and every element of claim 1, it is respectfully asserted that no *prima facie* case of anticipation of claim 1 has been made out. Claims 4-8, 11-13, 16, 22, and 38 are patentable over Krishnaswamy at least by virtue of their dependance from claim 1. Accordingly, the rejection of claims 1, 4-8, 11-13, 16, 22, and 38 should be reversed.

B. The Combination of Krishnaswamy and Klapproth is Legally Deficient to Establish a Prima Facie Case of Obviousness Against the Claimed Inventions

In rejecting claims under 35 U.S.C. 103, the Examiner bears the initial burden of presenting a prima facie case of obviousness. In re Rijckaert, 9 F.3d 1531, 1532 (Fed. Cir. 1993). The burden of presenting a prima facie case of obviousness is only satisfied by showing some objective teaching in the prior art or that knowledge generally available to one of ordinary skill in the art would lead that individual to combine the relevant teachings of the references. In re Fine, 837 F.2d 1071, 1074 (Fed. Cir. 1988). The test for obviousness is what the combined teachings of the applied prior art references would have suggested to one of ordinary skill in the art. In re Keller, 642 F.2d 413, 435; 208 U.S.P.Q. 871, 881 (CCPA 1981). The suggestion to combine the references should come from the prior art, and the Examiner cannot use hindsight gleaned from the invention itself to pick and choose among related prior art references to arrive at the claimed invention. In re Fine, 837 F.2d at 1075. If the Examiner fails to establish a prima facie case, the rejection is improper and must be overturned. In re Rijckaert, 9 F.3d at 1532 (citing In re Fine, 837 F.2d at 1074).

Here, Applicants respectfully assert that the combination of Krishnaswamy and Klapproth is legally deficient to establish a prima facie case of obviousness against any of claims 1, 4-8, 11-13, 16, 22, and 38. At the very least, independent claim 1 is not obvious over the combination of Krishnaswamy and Klapproth for the following reasons.

(i). The combination of Krishnaswamy and Klapproth fails to teach or suggest "storing, in a memory array, profile counts for a plurality of events associated with the execution of the computer program, the memory array being separate and distinct from the memory hierarchy so as to not perturb normal operations of the memory hierarchy," as claimed in claim 1.

On page 4 of the Office Action mailed on October 6, 2006, the examiner states that "Klapproth demonstrated that it was known at the of invention to provide a separate and distinct memory for tracing and profiling". However, more specifically, Klapproth disclosed, "an internal event trace buffer memory for accomodating a restricted set of contents of non-sequential addresses as generated by the processing element and allowing at least one of the following storage modes for a limited time operation of said microprocessor: storing of all non-sequential addresses, and/or storing of all call, jump, and trap addresses" (column 2, lines 41-48). This buffer memory is quite restricted and was not designed to store profile counts for a plurality of events associated with the execution of a program.

Furthermore, the examiner states that "[i]t would have been obvious to one of ordinary skill in the art at the time of invention to implement trace/profiling system of Krishnaswamy with a separate buffer for storing as found in Klapproth's teaching" because "one of ordinary skill in the art would have been motivated to provide an interface that doesn't unduly burden the bus".

However, based on the disclosure of Klapproth, it does not appear obvious or clear for that matter to make such a substitution. Figure 2 of Klapproth discloses an on-chip trace memory 58 embedded within a DSU 56, the DSU embedded within a microcontroller 20, the microcontroller embedded within a target board which is connected to a JTAG interface that is connected to the host workstation 32. It is neither clear nor obvious why one would want to piece together an on-chip event memory trace buffer restricted for the purposes discussed above and buried under so many layers and levels of circuit indirection with Krishnaswamy. The Examiner appears to be impermissibly using the hindsight gleaned from the invention itself to pick and choose among related disclosures in the prior art to arrive at the claimed invention.

Because the combination of Krishnaswamy and Klapproth neither teaches nor suggests each and every element of claim 1, it is respectfully asserted that no *prima facie* case of obviousness of claim 1 has been made out. Claims 4-8, 11-13, 16, 22, and 38 are patentable over the combination of Krishnaswamy and Klapproth at least by virtue of their dependance from claim 1. Accordingly, the rejection of claims 1, 4-8, 11-13, 16, 22, and 38 should be reversed.

C. The Combination of Krishnaswamy and Dictionary is Legally Deficient to Establish a Prima Facie Case of Obviousness Against the Claimed Inventions

Applicants respectfully assert that the combination of Krishnaswamy and Dictionary is legally deficient to establish a prima facie case of obviousness against any of claims 3, 9-10, 23-30, 32-34, 37, and 39. At the very least, independent claim 23 is not obvious over the combination of Krishnaswamy and Dictionary for the following reasons.

(i). The combination of Krishnaswamy and Dictionary fails to teach or suggest "a controller adapted to select the events for profiling," as claimed in claim 23.

The Examiner applies the same rejection applied for "selecting at least one of the plurality of events for profiling," as claimed in claim 1, for "a controller adapted to select the events for profiling," as claimed in claim 23. Appellants submit that the arguments provided in section (A)(i) above apply similarly to this section. In particular, that the counters in Krishnaswamy are "programmable" does not necessarily imply that a selection of the events is made.

Because the combination Krishnaswamy and Dictionary neither teaches nor suggests each and every element of claim 23, it is respectfully asserted that no *prima facie* case of obviousness of claim 23 has been made out. Claims 24-30 and 32-39 are patentable over the combination of Krishnaswamy and Dictionary at least by virtue of their dependance from claim 23. Accordingly, the rejection of claims 23-30 and 32-39 should be reversed.

(ii). The combination of Krishnaswamy and Dictionary fails to teach or suggest "a controller adapted...to update the profile counts of the selected events stored in said memory array" and "wherein the computer processing system assists compilation of the computer program, based upon the profile counts stored in the memory array," as claimed in claim 23.

Because the combination of Krishnaswamy and Dictionary does not teach or suggest "selecting at least one of the plurality of events for profiling," as shown in part (C)(i) above, it logically follows that the combination of Krishaswamy and Dictionary does not disclose "a controller adapted...to update the profile counts of the selected events stored in said memory array" and "wherein the computer processing system assists

compilation of the computer program, based upon the profile counts stored in the memory array."

Because the combination Krishnaswamy and Dictionary neither teaches nor suggests each and every element of claim 23, it is respectfully asserted that no *prima facie* case of obviousness of claim 23 has been made out. Claims 24-30 and 32-39 are patentable over the combination of Krishnaswamy and Dictionary at least by virtue of their dependance from claim 23. Accordingly, the rejection of claims 23-30 and 32-39 should be reversed.

(iii). The combination of Krishnaswamy and Dictionary fails to teach or suggest "a scaling circuit adapted to scale the profile counts to prevent profile information overflow," as claimed in claim 23 and essentially as claimed in claim 10.

The Examiner cites Dictionary as disclosing "a scaling circuit adapted to scale the profile counts to prevent profile information overflow," as claimed in claim 23 and as essentially claimed in claim 10. A generic definition of scaling does not provide for a scaling circuit or scaling functionality in the claims. Further, there is no clearer example of hindsight reconstruction than the one provided by the Examiner here. Nothing in Krishnaswamy or Dictionary indicate any motivation, teaching or suggestion for such a combination. The Examiner in hindsight merely found a definition in Dictionary to satisfy his rejection. It is impermissible to use the claimed invention as an instruction manual or "template" in attempting to piece together isolated disclosures and teachings of the prior art so that the claimed invention is rendered obvious. *In re Fritch*, 972 F.2d 1260, 1266, 23 USPQ2d 1780, 1783 (Fed. Cir. 1992). There is simply no other way to explain the disparate citation to Dictionary other than improper hindsight reasoning.

The Examiner states, without support, that it "would have been obvious [to combine Krishnaswamy with Dictionary] because one of ordinary skill in the art would be motivated to adjust the stored value to the hardware/equipment (register size limitations) (Computing, page 432)." It is entirely unclear to Appellants how the citation to page 432 of Dictionary supports the Examiner's statement. It is further unclear where such reasoning originated, other than from the Examiner's own imagination. The statement by the Examiner is clearly conclusory and without merit.

Because the combination Krishnaswamy and Dictionary neither teaches nor suggests each and every element of claim 23, it is respectfully asserted that no *prima facie* case of obviousness of claim 23 has been made out. Claims 24-30 and 32-39 are patentable over the combination of Krishnaswamy and Dictionary at least by virtue of their dependance from claim 23. Claims 3 and 9-10 are patentable over the combination of Krishnaswamy and Dictionary at least by virtue of their dependance from claim 1. Accordingly, the rejection of claims 3, 9-10, 23-30 and 32-39 should be reversed.

D. The Combination of Krishnaswamy, Klapproth and Dictionary is Legally Deficient to Establish a Prima Facie Case of Obviousness Against the Claimed Inventions

Applicants respectfully assert that the combination of Krishnaswamy, Klapproth and Dictionary is legally deficient to establish a prima facie case of obviousness against any of claims 3, 9-10, 23-30, 32-34, 37, and 39. At the very least, independent claim 23 is not obvious over the combination of Krishnaswamy, Klapproth and Dictionary for the following reasons.

(i). The combination of Krishnaswamy, Klapproth, and Dictionary fails to teach or suggest "storing, in a memory array, profile counts for a plurality of events associated with the execution of the computer program, the memory array being separate and distinct from the memory hierarchy so as to not perturb normal operations of the memory hierarchy," as claimed in claim 23 as essentially claimed in claim 10.

The Examiner applies the same rejection applied for a "memory array being separate and distinct from the memory hierarchy," as claimed in claim 1, for "memory array being separate and distinct from the memory hierarchy," as claimed in claim 23. Appellants submit that the arguments provided in section (B)(i) above apply similarly to this section. In particular, that it would not have been obvious to combine an event trace buffer memory from Klapproth with Krishnaswamy .

Because the combination Krishnaswamy, Klapproth and Dictionary neither teaches nor suggests each and every element of claim 23, it is respectfully asserted that no *prima facie* case of obviousness of claim 23 has been made out. Claims 24-30 and 32-39 are patentable over the combination of Krishnaswamy, Klapproth and Dictionary at least by virtue of their dependance from claim 23. Claims 3 and 9-10 are patentable over the combination of Krishnaswamy and Dictionary at least by virtue of their dependance from claim 1. Accordingly, the rejection of claims 3, 9-10, 23-30 and 32-39 should be reversed.

E. The Combination of Krishnaswamy and Chang is Legally Deficient to Establish a Prima Facie Case of Obviousness Against the Claimed Inventions

Applicants respectfully assert that the combination of Krishnaswamy, and Chang is legally deficient to establish a prima facie case of obviousness against any of claims 40-42.

At the very least, independent claim 40 is not obvious over the combination of Krishnaswamy and Chang for the following reasons.

(i). The combination of Krishnaswamy and Chang fails to teach or suggest "selecting at least one of the plurality of event-specific profile counts for profiling the path of the computer program," as claimed in claim 40.

The Examiner applies the same rejection applied for "selecting at least one of the plurality of events for profiling," as claimed in claim 1, for "selecting at least one of the plurality of event-specific profile counts for profiling the path of the computer program" as claimed in claim 40. Appellants submit that the arguments provided in section (A)(i) above apply similarly to this section. In particular, that the counters in Krishnaswamy are "programmable" does not necessarily imply that a selection of the events is made.

Because the combination of Krishnaswamy and Chang neither teaches nor suggests each and every element of claim 40, it is respectfully asserted that no *prima facie* case of obviousness of claim 40 has been made out. Claims 41-42 are patentable over the combination of Krishnaswamy and Chang at least by virtue of their dependence from claim 40. Accordingly, the rejection of claims 40-42 should be reversed.

(ii). The combination of Krishnaswamy and Chang fails to teach or suggest "if at least one of the selected event-specific profile counts has exceeded a predefined threshold, optimizing the portions of the computer program associated with the event-specific profile counts more aggressively than other portions of the computer program," as claimed in claim 40.

Because the combination of Krishnaswamy and Chang does not teach or suggest "selecting at least one of the plurality of event-specific profile counts for profiling the path of the computer program," as shown in part (E)(i) above, it logically follows that the

combination of Krishaswamy and Chang does not disclose "if at least one of *the selected event-specific profile counts* has exceeded a predefined threshold, optimizing the portions of the computer program associated with the event-specific profile counts more aggressively than other portions of the computer program."

Because the combination of Krishnaswamy and Chang neither teaches nor suggests each and every element of claim 40, it is respectfully asserted that no *prima facie* case of obviousness of claim 40 has been made out. Claims 41-42 are patentable over the combination of Krishnaswamy and Chang at least by virtue of their dependence from claim 40. Accordingly, the rejection of claims 40-42 should be reversed.

(iii). No proper suggestion, teaching or motivation is provided for the combination of Krishnaswamy and Chang.

In combining Krishnaswamy and Chang, the Examiner states, without support, that the combination "would have been obvious because one of ordinary skill in the art would be motivated to optimize frequently executed program paths primarily since they are executed more (page 1301, Introduction and pages 1306-1308)." The citation to Chang does not provide any motivation or suggest to combine the optimizations of Chang with the PMU counters of Krishnaswamy. The only plausible support for combining Krishnaswamy and Chang in such a manner is improper hindsight reasoning using the Appellants' disclosure.

Because no proper motivation, teaching or suggstion is provided for the combination of Krishnaswamy and Chang, it is respectfully asserted that no *prima facie* case of obviousness of claim 40 has been made out. Claims 41-42 are patentable over the

combination of Krishnaswamy and Chang at least by virtue of their dependence from claim 40. Accordingly, the rejection of claims 40-42 should be reversed.

F. The Combination of Krishnaswamy, Klapproth and Chang is Legally Deficient to Establish a Prima Facie Case of Obviousness Against the Claimed Inventions

Applicants respectfully assert that the combination of Krishnaswamy, Klapproth, and Chang is legally deficient to establish a prima facie case of obviousness against any of claims 40-42. At the very least, independent claim 40 is not obvious over the combination of Krishnaswamy, Klapproth and Chang for the following reasons.

(i). The combination of Krishnaswamy, Klapproth and Chang fails to teach or suggest "storing, in a memory array, a plurality of event-specific profile counts, each associated with an event associated with the execution of a path of the computer program, the memory array being separate and distinct from the memory hierarchy so as to not perturb normal operations of the memory hierarchy," as claimed in claim 40.

The Examiner applies the same rejection applied for a "memory array being separate and distinct from the memory hierarchy," as claimed in claim 1, for "memory array being separate and distinct from the memory hierarchy," as claimed in claim 40. Appellants submit that the arguments provided in section (B)(i) above apply similarly to this section. In particular, that it would not have been obvious to combine an event trace buffer memory from Klapproth with Krishnaswamy .

Because the combination of Krishnaswamy, Klapproth and Chang neither teaches nor suggests each and every element of claim 40, it is respectfully asserted that no *prima facie* case of obviousness of claim 40 has been made out. Claims 41-42 are patentable over

the combination of Krishnaswamy and Chang at least by virtue of their dependence from claim 40. Accordingly, the rejection of claims 40-42 should be reversed.

G-P. Claims 14-15, 18-19, 20-21, 35-36, and 41-41 are patentable by virtue of their dependence from their respective base claims.

This section addresses grounds of rejection G-P. All such grounds are over dependent claims which are believed to be patentable by virtue of their dependence from their base claims. The patentability of said base claims were addressed above in sections A-F. Claims 14-15, 18-19, 20-21 are believed to be patentable by virtue of their dependence from claim 1. Claims 35-36 are believed to be patentable by virtue of their dependence from claim 23. Claims 41-42 are believed to be patentable by virtue of their dependence from claim 40. Accordingly, the rejection of claims 14-15, 18-19, 20-21, 35-36, and 40-42 should be reversed.

Q. CONCLUSION

Accordingly, for at least the above reasons, it is respectfully requested that the Board reverse all claim rejections under 35 U.S.C. § 102 and 103.

Respectfully submitted,

By: _____

Frank Chau
Reg. No. 34,136
Attorney for Appellants

F. Chau & Associates, LLC
130 Woodbury Road
Woodbury, NY 11797
TEL: (516) 692-8888
FAX: (516) 692-8889
Attorneys for Appellants

Claims Appendix

1. A method for profiling computer program executions in a computer processing system having a processor and a memory hierarchy, comprising the steps of:

executing a computer program;

storing, in a memory array, profile counts for a plurality of events associated with the execution of the computer program, the memory array being separate and distinct from the memory hierarchy so as to not perturb normal operations of the memory hierarchy;

selecting at least one of the plurality of events for profiling;

updating the profile counts for only the selected events;

assisting compilation and optimization of the computer program, based upon the selected profile counts stored in the memory array.

2. (Cancelled).

3. The method according to claim 1, wherein said storing and updating steps are performed asynchronously to prevent a decrease of an execution speed of the computer program.

4. The method according to claim 1, wherein said updating step is triggered by execution of the events.

5. The method according to claim 1, wherein said updating step is triggered by execution of instructions embedded into an instruction stream of the computer program.

6. he method according to claim 1, further comprising the step of detecting whether a profile count has exceeded an adjustable predefined threshold.

7. The method according to claim 1, further comprising the step of indicating when a profile count has exceeded an adjustable predefined threshold.

8. The method according to claim 7, wherein said indicating step comprises the step of raising an exception.

9. The method according to claim 1, further comprising the steps of:
accumulating profile updates; and
dividing the accumulated profile updates by a threshold fraction.

10. The method according to claim 1, further comprising the step of scaling the profile counts to prevent profile information overflow.

11. The method according to claim 1, further comprising the step of identifying profile information corresponding to the profile counts using a profiling event identifier.

12. The method according to claim 11, further comprising the step of addressing the memory array, using the profiling event identifier.

13. The method according to claim 1, further comprising the steps of:
generating the profile counts using profile counters associated with the events; and
maintaining the profile counters in a set-associate manner.

14. The method according to claim 13, further comprising the step of selecting a profile counter to be evicted from the memory array based upon a predefined replacement, when a number of profiling events assigned to an associative class of events is exceeded.

15. The method according to claim 14, wherein the replacement strategy is based upon one of least-recently-used and first-in-first-out.

16. The method according to claim 1, further comprising the step of supporting read operations from the memory array in an off-line optimization of the program.

17. (Cancelled).

18. The method according to claim 1, wherein said assisting step is performed during at least one of dynamic binary translation and dynamic optimization of the computer program.

19. The method according to claim 18, wherein the dynamic binary translation and dynamic optimization of the computer program results in translated and optimized code, respectively, the translated and optimized code comprising instructions groups which pass control therebetween.

20. The method according to claim 19, further comprising the step of identifying frequently executed paths of the computer program, by instrumenting exits from the instruction groups with a profiling instruction that indicates a unique group exit identifier.

21. The method according to claim 19, further comprising the step of extending the instruction groups along a frequently executed path.

22. The method according to claim 1, wherein the memory hierarchy includes data and instruction caches, and the memory array is separate and distinct from the memory hierarchy so as to not perturb normal operations of the data and instruction caches.

23. An apparatus for profiling computer program executions in a computer processing system having a processor and a memory hierarchy, the apparatus comprising:

a memory array adapted to store profile counts for events associated with execution of the computer program, said memory array being separate and distinct from the memory hierarchy so as to not perturb normal operations of the memory hierarchy;

a controller adapted to select the events for profiling and to update the profile counts of the selected events stored in said memory array; and

a scaling circuit adapted to scale the profile counts to prevent profile information overflow;

wherein the computer processing system assists compilation of the computer program, based upon the profile counts stored in the memory array.

24. The apparatus according to claim 23, wherein said memory array and said controller are adapted to asynchronously store and update the profile counts, respectively, to prevent a decrease of an execution speed of the computer program.

25. The apparatus according to claim 23, wherein said controller is adapted to update the profile counts as the events are executed.

26. The apparatus according to claim 23, wherein said controller is adapted to update the profile counts based upon instructions embedded into an instruction stream of the computer program.

27. The apparatus according to claim 23, further comprising a comparator circuit adapted to detect whether a profile count has exceeded an adjustable predefined threshold.

28. The apparatus according to claim 23, further comprising an indicating circuit for indicating when a profile count has exceeded an adjustable predefined threshold.

29. The apparatus according to claim 28, wherein said indicating circuit is adapted to raise an exception when the profile count has exceeded the adjustable predefined threshold.

30. The apparatus according to claim 23, further comprising:

an accumulation circuit adapted to accumulate the updated profile counts; and

a dividing circuit adapted to divide an accumulated value of the updated accumulated profile counts by a threshold fraction.

31. (Cancelled).

32. The apparatus according to claim 23, wherein profile information corresponding to the profile counts is identified using a profiling event identifier.

33. The apparatus according to claim 32, wherein the memory array is addressed using the profiling event identifier.

34. The apparatus according to claim 23, further comprising profile counters for generating the profile counts, said profile counters being associated with an event in a set-associate manner.

35. The apparatus according to claim 34, further comprising a replacement circuit adapted to select a profile counter to be evicted from the memory array based on a predefined replacement strategy, when a number of profiling events assigned to an associative class is exceeded.

36. The apparatus according to claim 35, wherein the predefined replacement strategy is based upon one of least-recently-used and first-in-first-out.

37. The apparatus according to claim 23, wherein the memory hierarchy includes data and instruction caches, and said memory array is separate and distinct from the memory hierarchy so as to not perturb normal operations of the data and instruction caches.

38. The method according to claim 1, wherein said method is implemented by a program storage device readable by machine, tangibly embodying a program of instructions executable by the machine to perform said method steps.

39. The apparatus according to claim 23, further comprising wherein the computer processing system assists optimization of the computer program, based upon the profile counts stored in the memory array.

40. A method for profiling computer program executions in a computer processing system having a processor and a memory hierarchy, comprising the steps of:

executing a computer program;

storing, in a memory array, a plurality of event-specific profile counts, each associated with an event associated with the execution of a path of the computer program, the memory array being separate and distinct from the memory hierarchy so as to not perturb normal operations of the memory hierarchy;

selecting at least one of the plurality of event-specific profile counts for profiling the path of the computer program; and

if at least one of the selected event-specific profile counts has exceeded a predefined threshold, optimizing the portions of the computer program associated with the event-specific profile counts more aggressively than other portions of the computer program.

41. The method according to claim 40, further comprising the step of optimizing of the portions of the computer program during at least one of static and dynamic compilation.

42. The method according to claim 40, wherein the memory array is arranged as a two-way set associative array.

Evidence Appendix

None

Related Procedings Appendix

None